Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Spike and slab biclustering

M. Denitto[a,*], M. Bicego[a], A. Farinelli[a], M.A.T. Figueiredo[b]

[a] University of Verona, Strada Le Grazie 15, Ca' Vignal 2, Verona, Italy
[b] Instituto de Telecomunicações, and Instituto Superior Técnico, Universidade de Lisboa, Av. Rovisco Pais 1, Lisboa, Portugal

ARTICLE INFO

Article history:
Received 14 September 2016
Revised 29 May 2017
Accepted 19 July 2017
Available online 20 July 2017

Keywords:
Biclustering
Spike and slab
Probabilistic graphical models
Expectation-maximization

ABSTRACT

Biclustering refers to the problem of simultaneously clustering the rows and columns of a given data matrix, with the goal of obtaining submatrices where the selected rows present a coherent behaviour in the selected columns, and vice-versa. To face this intrinsically difficult problem, we propose a novel generative model, where biclustering is approached from a sparse low-rank matrix factorization perspective. The main idea is to design a probabilistic model describing the factorization of a given data matrix in two other matrices, from which information about rows and columns belonging to the sought for biclusters can be obtained. One crucial ingredient in the proposed model is the use of a spike and slab sparsity-inducing prior, thus we term the approach *spike and slab biclustering* (SSBi). To estimate the parameters of the SSBi model, we propose an *expectation-maximization* (EM) algorithm, termed SSBiEM, which solves a low-rank factorization problem at each iteration, using a recently proposed augmented Lagrangian algorithm. Experiments with both synthetic and real data show that the SSBi approach compares favorably with the state-of-the-art.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

The goal of *biclustering*, or co-clustering, is to simultaneously cluster both rows and columns of a given data matrix, such that the resulting sub-matrices exhibit some form of coherence. The term biclustering was coined in the microarray gene expression analysis context [1–5], with the goal of identifying groups of genes that show similar activity patterns in a subset of the experimental conditions, thus potentially revealing novel biological information. Unlike classical one-way clustering, biclustering can reveal crucial information in the microarray scenario for the following reasons: (i) only a small set of the genes may be involved in some cellular process of interest; (ii) the cellular process of interest may be active only in a subset of the conditions; (iii) a single gene may participate in multiple processes that may, or not, be co-active under all conditions. Although biclustering has been mostly used with biological data, recent years saw it being successfully applied in other areas, such as market segmentation and data mining [6–11].

Several different approaches have been proposed for biclustering, each characterized by different features, such as accuracy, computational complexity, descriptiveness of the retrieved biclusters, as reviewed comprehensively in [2] and [3]. Most biclustering methods can be divided into four main classes [3]:

1. **correlation maximization methods**, as suggested by their name, seek biclusters that maximize the correlation between the rows and the columns belonging to them [1,12];
2. **variance minimization methods** search for biclusters minimizing the row variance along the columns belonging to the biclusters, or vice-versa [13–15];
3. **two-way clustering methods** retrieve biclusters by alternating between row and column clustering (examples of this class are found in [16–19]);
4. **probabilistic/generative methods** exploit probabilistic models and inference tools to retrieve rows coherently expressed in a columns subsets (examples include the methods proposed in [20–24]).

In this paper, we focus on and contribute to the latter group. Although probabilistic approaches tend to be computational heavy, they offer several advantages. The underlying *generative model* (*i.e.*, the probabilistic mechanism assumed to be behind the data) can be used to generate synthetic data, which when compared with real data, allows assessing the validity of the model. Moreover, probabilistic inference provides estimates of the confidence/uncertainty level of the obtained estimates. Different probabilistic approaches to biclustering have been proposed. In [21], for example, each bicluster begins as a seed that is iteratively optimized by adding/removing rows and columns to/from the cluster by sampling from a conditional probability distribution using a *Monte Carlo* procedure. That iterative procedure is akin to

* Corresponding author.
E-mail address: matteo.denitto@univr.it (M. Denitto).

http://dx.doi.org/10.1016/j.patcog.2017.07.021
0031-3203/© 2017 Elsevier Ltd. All rights reserved.

a *Markov chain Monte Carlo* (MCMC) process. Another interesting probabilistic approach to biclustering was presented in [22], where the authors tackle biclustering using a strategy based on a simple frequency model for the expression pattern of a bicluster and on *Gibbs sampling* for parameter estimation. Alternatively, in [25], the authors proposed a graph-theoretic approach aiming at finding maximum bounded bi-cliques through a statistical representation of the data. Another notable example is the one developed in [24], where the authors face biclustering from a geometric point of view, to retrieve sub-biclusters, which are then combined into larger ones, using a probabilistic relaxation labeling framework.

A recently proposed generative multiplicative model, based on sparse matrix factorization and known as FABIA (*factor analysis for biclustering acquisition* [20]) overcomes some limitations of previous probabilistic techniques. Whereas previous methods used additive models under a Gaussianity assumption (which is known not to be true, since the pre-processing of micro-array data may yield heavy-tailed distributions [20,26]), FABIA is based on factor analysis with a sparsity prior on the elements of the factors. The basic idea is to decompose the data matrix in levels, each corresponding to a different bicluster. Although the performance of FABIA is very promising, the corresponding likelihood is not analytically tractable, so its authors resorted to a *variational expectation-maximization* (VEM) algorithm to estimate the model parameters [20]. Another drawback of FABIA is that the model does not provide information about bicluster memberships, hence requiring post-processing of its output to obtain that information [20].

In this paper, we add a new member to the family of probabilistic biclustering methods, by proposing a novel generative model, which we call *spike and slab biclustering* (SSBi).[1] The proposed SSBi formulation approaches biclustering from a *probabilistic sparse low-rank matrix factorization* perspective. Similarly to FABIA, SSBi works by factoring the data matrix into the product of two matrices that provide information about the underlying biclusters. The proposed method involves two main ingredients.

1. The data matrix is approximated by a *low rank* matrix; in particular, each bicluster has rank 1 and corresponds to the outer multiplication of two vectors, with the data matrix being modeled as the sum of a collection of such rank-1 products (*i.e.*, biclusters).
2. The vectors that correspond to each bicluster are expected to be *sparse*; in fact, most data matrices addressed in biclustering work have a large number of rows/columns (*i.e.*, thousands by hundreds, in gene expression data) and the biclusters typically involve only small portions thereof.

Since, in practice, no matrix of real data is exactly low rank, we model deviations from the low rank assumption as a Gaussian perturbation added to the underlying low-rank matrix. In order to enforce sparsity on the factors, we propose to use a *spike and slab* prior. The original spike and slab was proposed by Mitchell and Beauchamp [28] for variable selection in linear regression and later generalized and adopted by many authors as a general-purpose sparsity-inducing prior [29]. In its basic form, the spike and slab is a univariate prior composed by the mixture of two zero-mean Gaussian distributions: one with very small variance, modeling a high probability of nearly zero values, and another one with large variance, which models the presence a large values (see Fig. 1).

[1] Very recently, an approach sharing similar ideas to those presented in this paper was proposed in the specific context of NCI-DREAM drug sensitivity prediction [27]. That paper proposes using a spike and slab prior in the generalized factor analysis previously introduced. However, that work uses a different member of the spike and slab family and, moreover, the learning algorithm adopted is significantly different: we use expectation-maximization, while they use Gibbs sampling.

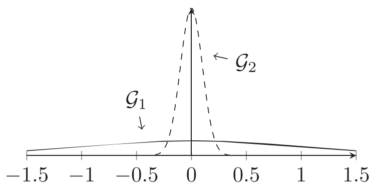

Fig. 1. Example of spike and slab prior distribution. The low variance Gaussian $\mathcal{G}_2 = \mathcal{N}(0, 0.1)$ describes nearly zero samples, while the large variance Gaussian $\mathcal{G}_1 = \mathcal{N}(0, 1)$ models large magnitude values.

In contrast with FABIA, the proposed SSBi formulation leads to a computationally tractable likelihood, allowing us to estimate the proposed generative model parameters through an instance of the *expectation-maximization* (EM) algorithm, which we refer to as SSBiEM. The SSBiEM algorithm exploits a recently proposed augmented Lagrangian method for low-rank factorization. Moreover, SSBiEM directly produces bicluster membership information, thus dispensing with the need for any post-processing step.

The proposed SSBiEM method was experimental evaluated on both synthetic data (the FABIA synthetic benchmark [20]) and real data (the Breast Tumor dataset [7]). The results reported below show that SSBiEM improves over the results of FABIA, and compares favorably with other state-of-the-art approaches.

The remainder of the paper is organized as follows. Section 2 provides a brief review of biclustering and sparse low-rank matrix factorization. Section 3 presents the proposed SSBi model, while Section 4 describes SSBiEM. Section 5 presents and discusses the experimental evaluation. Finally, some concluding remarks are presented in Section 6.

Notation: we refer to matrices using capital letters (*e.g., D, V, Z*), to vectors with lower-case letters (*e.g., d, v, z*), and to matrix/vector elements using subscripts (*e.g.*, the entry (i, j) of matrix A is a_{ij} and the component p of vector d is d_p). The so-called "vec" operator (vectorization) takes a matrix argument and returns a vector with the matrix elements stacked column by column. The reverse operation is denoted vec^{-1} (*i.e.*, such that $\text{vec}^{-1}\big(\text{vec}(A)\big) = A$). A pair of useful equalities concerning the vec operator are

$$\text{vec}(AB) = (I \otimes A)\text{vec}(B) = (B^T \otimes I)\text{vec}(A), \tag{1}$$

where I is an identity matrix of adequate dimensions and $\otimes$ is the Kronecker matrix product [30]. Finally, given some matrix A, $\|A\|_F$ denotes its Frobenius norm, which is the Euclidean norm of its vectorization: $\|A\|_F = \|\text{vec}(A)\|_2$.

2. Background

This section provides background knowledge underlying the proposed method, namely the biclustering problem and the current state-of-the-art regarding biclustering via sparse low-rank matrix factorization.

2.1. Biclustering and sparse low-rank factorization

As mentioned in Section 1, biclustering aims at the simultaneous clustering of rows and columns of a given data matrix. We denote as $D \in \mathbb{R}^{n\times m}$ the given data matrix, and let $R = \{1, \ldots, n\}$ and $C = \{1, \ldots, m\}$ be the set of row and column indices. We adopt D_{TK}, where $T \subseteq R$ and $K \subseteq C$, to represent the submatrix with the subset of rows in T and the subset of columns in K. Given this notation, we can define a *bicluster* as a submatrix D_{TK}, such that the subset of rows of D with indices in T exhibits a "coherent behavior" (in some sense) across the set of columns with indices in K, and *vice versa*. The choice of coherence criterion defines the type of biclusters to be retrieved (for a comprehensive survey of biclustering criteria, see [2,3]).

A possible coherence criterion for a bicluster (sub-matrix) is for the corresponding entries to share the same pattern, significantly different from the other entries of the matrix. In what follows, we present two examples presenting different types of biclusters on two data matrices D_1 and D_2. In the first example, the bicluster corresponds to the subset of rows $T_1 = \{1, 2, 4, 5\}$ and the subset of columns $K_1 = \{1, 3\}$; in the second example, the bicluster corresponds to the subset of rows $T_2 = \{1, 2, 3, 4\}$ and the subset of columns $K_2 = \{1, 4\}$. The matrices are as follows:

$$D_1 = \begin{bmatrix} \mathbf{10} & 0 & \mathbf{20} & 0 \\ \mathbf{10} & 0 & \mathbf{20} & 0 \\ 0 & 0 & 0 & 0 \\ \mathbf{10} & 0 & \mathbf{20} & 0 \\ \mathbf{10} & 0 & \mathbf{20} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad D_2 = \begin{bmatrix} \mathbf{10} & 0 & 0 & \mathbf{20} \\ \mathbf{20} & 0 & 0 & \mathbf{40} \\ \mathbf{30} & 0 & 0 & \mathbf{60} \\ \mathbf{40} & 0 & 0 & \mathbf{80} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

From an algebraic point of view, these matrices can be represented by outer products of sparse vectors, $D_1 = v_1 z_1^T$ and $D_2 = v_2 z_2^T$, where

$$v_1 = \begin{bmatrix} \mathbf{5} \\ \mathbf{5} \\ 0 \\ \mathbf{5} \\ \mathbf{5} \\ 0 \end{bmatrix}, \quad z_1 = \begin{bmatrix} \mathbf{2} \\ 0 \\ \mathbf{4} \\ 0 \end{bmatrix}, \quad v_2 = \begin{bmatrix} \mathbf{5} \\ \mathbf{10} \\ \mathbf{15} \\ \mathbf{20} \\ 0 \\ 0 \end{bmatrix}, \quad z_2 = \begin{bmatrix} \mathbf{2} \\ 0 \\ 0 \\ \mathbf{4} \end{bmatrix}.$$

As in [20], we call these vectors *prototypes* (for v) and *factors* (for z). Generalizing to k biclusters, we can formulate the biclustering problem as the decomposition of the given data matrix D as the sum of k outer products,

$$D = \sum_{i=1}^{k} v_i z_i^T = VZ, \tag{2}$$

where $V = [v_1, \ldots, v_k] \in \mathbb{R}^{n\times k}$ and $Z = [z_1, \ldots, z_k]^T \in \mathbb{R}^{k\times m}$.

The connection between biclustering and sparse low-rank matrix factorization can be evidenced by observing that the factorization of the original data matrix shows that it has rank no larger than to the number of biclusters (usually much lower than the number of rows or columns). Moreover, if the size of matrix D is much bigger than the bicluster size (as it is typically the case in many applications), the resulting prototype and factor vectors should be composed mostly by zeros (*i.e.*, the prototypes and factors should be sparse). Note that if two entries in the approximated matrix belong to the same bicluster ($D_{ij} \approx D_{il}$ for constant valued biclusters as in the examples), it is not strictly required that prototypes and factors approximating them should have similar sparse constraints. In fact, what should be similar is the sum of the products in Eq. (2).

In the literature, there are several proposals of biclustering methods based on matrix factorization [12,20,31]. A recent trend in this context is to use non-negative matrix tri-factorisation to tackle the objective of biclustering [32–34]. Specifically, those techniques require the obtained matrices to be orthogonal. Consequently, and crucially different from what we present, the retrieved biclusters cannot overlap. This could represent a huge limitation in contexts where biclustering is commonly applied, such as for gene expression analysis where the same gene can participate in several biological processes. Another significant class of approaches relies on what are known as *latent block models* [35–38]; differently from our proposal, the goal of such approaches is to simultaneously rearrange rows and columns into groups of similar response patterns. The common assumption made by these models is that each row or column belongs exclusively to one row or column group, respectively. Hence they do not allow for overlapping biclusters. Finally, similarly to what we propose, FABIA (discussed next) Hochreiter et al. [20] is a probabilistic matrix factorization techniques allowing for overlapping biclusters.

2.2. FABIA

FABIA is a generative model for biclustering based on factor analysis [20]. The model proposed to decompose the data matrix is obtained by adding noise to the strict low rank decomposition in (2),

$$D = \sum_{i=1}^{k} v_i z_i^T + Y = VZ + Y, \tag{3}$$

where matrix $Y \in \mathbb{R}^{n\times m}$ accounts for random noise or perturbations, assumed to be zero-mean Gaussian with a diagonal covariance matrix. As explained above (Sections 1 and 2.1) the protoypes in V and the factors in Z should be sparse. To induce sparsity, FABIA uses two type of priors: (i) an independent Laplacian prior, and (ii) a prior distribution that is non-zero only in region where prototypes are sparse (for further details, see [20]). This model formulation leads to an analytically intractable likelihood, preventing the derivation of exact forms for the steps of the EM algorithm. Because of that, the model parameters are estimated using a variational EM (VEM) algorithm [20,39].

Another important drawback of FABIA is the fact that no information about biclusters membership is explicitly encoded in the model. Thus, the authors of FABIA proposed a post-processing scheme to retrieve bicluster memberships, which is based on thresholds that need to be chosen and which critically affect the retrieved biclusters.

The approach proposed in this paper, described in the next section, overcomes all these drawbacks.

3. Spike and slab biclustering

This section contains a detailed description of the proposed model, which we call *spike and slab biclustering* (SSBi). We first review the spike and slab formulation and how it is instantiated to yield SSBi.

3.1. Spike and slab

The so-called spike and slab is a probabilistic model that has been successfully used as a prior for variable selection in linear regression and other problems [28,29]. Formally, the basic spike and slab prior is a mixture of two zero-mean Gaussians, one with a very small variance and the other with large variance. Under this density, both very large and very small (nearly zero) samples have high likelihood, something that is not possible under a single Gaussian. An illustration of a spike and slab prior is shown in Fig. 1. To generate a sample from this model, we begin by randomly selecting (with a certain probability) one of the two Gaussians, and then obtain a sample from the chosen distribution.

Formally, the spike and slab prior has the form

$$\mathcal{P}(x|\alpha, \tau_1, \tau_2) = \alpha\,\mathcal{N}(x|0, \tau_1^2) + (1-\alpha)\mathcal{N}(x|0, \tau_2^2), \tag{4}$$

with $\tau_2 \lll \tau_1$, parameter $0 \le \alpha \le 1$ controls the sparsity degree, and $\mathcal{N}(x|\mu, \sigma^2)$ denotes a Gaussian density with mean μ and variance σ^2, computed at x.

Note that (4) is equivalent to the following two-stage model

$$\mathcal{P}(x|h, \tau_1, \tau_2) = \mathcal{N}(x|0, \tau_1^2)^h \mathcal{N}(x|0, \tau_2^2)^{(1-h)}, \tag{5}$$

$$\mathcal{P}(h|\alpha) = \alpha^h (1-\alpha)^{1-h}, \tag{6}$$

where $h \in \{0, 1\}$ is a (not observed, or latent) binary variable following a Bernoulli distribution of parameter α. The mixture in (4) results from marginalizing this model with respect to h.

3.2. The SSBi model

There are two main ingredients in the proposed SSBi approach.

1. The data matrix D is modeled as in FABIA, *i.e.*, with a Gaussian distribution having the product VZ as mean and σ as standard deviation (representing the approximation noise). This part provides the "low-rank" assumption, since the approximation matrix has rank k, at maximum.
2. The prototype and factor matrices (*i.e.*, V and Z) are *sparse*, and we adopt a spike and slab prior to induce that feature.

The probabilistic graphical model describing the proposed approach (which is sketched in Fig. 2) is formally defined as follows.

- Given the product VZ, the entries of the data matrix D are i.i.d. Gaussian with variance σ^2:

$$\mathcal{P}(D|V,Z,\sigma^2) = \mathcal{N}(D|VZ,\sigma^2 I) = \prod_{i=1}^{n}\prod_{j=1}^{m}\mathcal{N}(d_{ij}|(VZ)_{ij},\sigma^2). \tag{7}$$

- The entries of V follow a spike and slab prior with variances τ_1^2 and τ_2^2 (such that $\tau_1^2 \gg \tau_2^2$),

$$\mathcal{P}(V|H,\tau_1,\tau_2) = \prod_{i=1}^{n}\prod_{j=1}^{k}\mathcal{N}(v_{ij}|0,\tau_1^2)^{h_{ij}}\mathcal{N}(v_{ij}|0,\tau_2^2)^{1-h_{ij}}, \tag{8}$$

  where the binary latent variables in matrix H follow a Bernoulli distribution of parameter α_1,

$$\mathcal{P}(H|\alpha_1) = \prod_{i=1}^{n}\prod_{j=1}^{k}\alpha_1^{h_{ij}}(1-\alpha_1)^{1-h_{ij}}. \tag{9}$$

- The entries of Z also follow a spike and slab prior, with variances ρ_1^2 and ρ_2^2 (such that $\rho_1^2 \gg \rho_2^2$),

$$\mathcal{P}(Z|G,\rho_1,\rho_2) = \prod_{i=1}^{k}\prod_{j=1}^{m}\mathcal{N}(z_{ij}|0,\rho_1^2)^{g_{ij}}\mathcal{N}(z_{ij}|0,\rho_2^2)^{1-g_{ij}}, \tag{10}$$

  where the binary latent variables in matrix G follow a Bernoulli distribution of parameter α_2,

$$\mathcal{P}(G|\alpha_2) = \prod_{i=1}^{k}\prod_{j=1}^{m}\alpha_2^{g_{ij}}(1-\alpha_1)^{1-g_{ij}}. \tag{11}$$

Intuitively α_1 and α_2 regulate the sparsity degree in each prototype and factor vector or, equivalently, the biclusters dimensions on rows and columns respectively. The standard deviations τ_1, τ_2, ρ_1 and ρ_2 control the value ranges.

The joint distribution of all the variables and parameters involved in this model can now be written as

$$\begin{aligned} &P(D,V,Z,H,G,\sigma,\tau_1,\tau_2,\rho_1,\rho_2,\alpha_1,\alpha_2) \\ &= \mathcal{P}(D|V,Z,\sigma^2)\mathcal{P}(V|H,\tau_1,\tau_2)\mathcal{P}(Z|G,\rho_1,\rho_2) \\ &\quad \mathcal{P}(H|\alpha_1)\mathcal{P}(G|\alpha_2)\mathcal{P}(\sigma,\tau_1,\tau_2,\rho_1,\rho_2,\alpha_1,\alpha_2), \end{aligned} \tag{12}$$

where $\mathcal{P}(\tau_1,\tau_2,\rho_1,\rho_2,\alpha_1,\alpha_2)$ is a prior on the model parameters. In this paper, we consider this prior to be flat, that is, we seek *maximum likelihood* (ML) estimates thereof.

Finally, notice that this model may be easily extended to the case where each bicluster has its own parameter set (the spike and slab variances and mixing probability), rather than being assumed the same for all the biclusters. To keep the notation simpler, we will proceed with the simpler version just introduced.

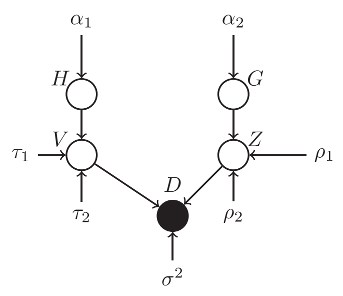

Fig. 2. The probabilistic graphical model of the proposed spike and slab biclustering. The corresponding conditionals are given by Eqs. (7)–(11). Please note that, although in the EM algorithm we treat V and Z as parameters, from a general perspective all V, Z, H and G are latent variables.

4. Parameter estimation

In this section, we propose an EM algorithm (herein called SSBiEM) to estimate all the parameters of the SSBi model proposed in the previous section. Recall that EM is a classical iterative algorithmic framework to obtain a marginal maximum likelihood estimate $\hat{\theta} = \arg\max_\theta \mathcal{P}(x|\theta)$, where the marginal likelihood results from marginalizing out a set of missing/hidden/latent variables y, *i.e.*, $\mathcal{P}(x|\theta) = \int \mathcal{P}(x,y|\theta)\,dy$ (with summation rather than integration, if y is discrete) [40]. The algorithm alternates between two steps:

E-step: computes the conditional expectation of the complete log-likelihood, given the current parameter estimate $\hat{\theta}^{(t)}$ and the observed data x, the so-called Q-function:

$$Q(\theta,\hat{\theta}^{(t)}) = \mathbb{E}_y\Big[\log\mathcal{P}(x,y|\theta)|x,\hat{\theta}^{(t)}\Big].$$

M-step: updates the parameter estimate by maximizing the Q-function:

$$\hat{\theta}^{(t+1)} = \arg\max_\theta Q(\theta,\hat{\theta}^{(t)}).$$

Computing the expectation yielding the Q-function may not be trivial in general, as it may involve intractable integration. This is the case, *e.g.*, in the FABIA model [20].

In the model herein proposed, the goal is to estimate the model parameters, τ_1, τ_2, ρ_1, ρ_2, α_1, α_2, and σ, given a data matrix D, and assuming a known number of biclusters k. Concerning the unobserved V, Z, H, and G, we have the choice of marginalizing them out, which can be done via the EM algorithm by treating them as latent variables, or maximizing with respect to them, which corresponds to seeing them as parameters rather than latent variables.

Inspired by [41], and in order to obtain a simpler E-Step, we treat H and G as hidden variables, but V and Z as unknown parameters to be estimated along with τ_1, τ_2, ρ_1, ρ_2, α_1, α_2, and σ. We could also treat H and G as parameters; however, since these are matrices of binary variables, maximize with respect to them would correspond to taking hard decisions, which may have a strong influence in the whole optimization procedure. On the other hand, V and Z are matrices of real-valued entries, thus estimating them has a smoother/weaker influence in the estimates of the other quantities. For these reasons, we define V and Z as parameters, and H and G as hidden variables.

In what follows, we present the form that the E-step and the M-step take in the proposed SSBi model.

4.1. The E-Step

To keep the notation more compact, we denote the complete set of parameters as $\theta = \{V,Z,\sigma^2,\alpha_1,\alpha_2,\tau_1,\tau_2,\rho_1,\rho_2\}$. Recall that the joint distribution of all the variables and parameters is given in (12). With D observed and H and G as latent, the $\mathcal{Q}$ function is obtained by computing

$$\mathcal{Q}(\theta, \hat{\theta}^{(t)}) = \mathbb{E}_{H,G}\Big[\log \mathcal{P}(D, H, G, \theta)\Big|\hat{\theta}^{(t)}, D\Big].$$

After straightforward, but long and tedious analytic manipulations, and dropping any terms that do not depend on θ, we obtain the following closed-form expression:

$$\begin{aligned}\mathcal{Q}(\theta, \hat{\theta}^{(t)}) = &-\frac{nm}{2}\log(\sigma^2) - \frac{||D - VZ||^2}{2\sigma^2}\\ &-\frac{||\overline{H}^{(t)}||_F}{2}\log(\tau_1^2) - \frac{||1-\overline{H}^{(t)}||_F}{2}\log(\tau_2^2)\\ &-\frac{||\overline{G}^{(t)}||_F}{2}\log(\rho_1^2) - \frac{||1-\overline{G}^{(t)}||_F}{2}\log(\rho_2^2)\\ &-\frac{1}{2}v^T\overline{H}^{(t)}v - \frac{1}{2}z^T\overline{G}^{(t)}z\\ &+\Big(\sum_{p=1}^{nk}\overline{h}_p^{(t)}\Big)\log\Big(\frac{\alpha_1}{1-\alpha_1}\Big) + nk\log(1-\alpha_1)\\ &+\Big(\sum_{j=1}^{km}\overline{g}_j^{(t)}\Big)\log\Big(\frac{\alpha_2}{1-\alpha_2}\Big) + km\log(1-\alpha_2)\end{aligned} \quad (13)$$

where $v = \mathrm{vec}(V)$, $z = \mathrm{vec}(Z)$,

$$\overline{H}^{(t)} = \mathrm{diag}\Big(\frac{\overline{h}_1^{(t)}}{\tau_1^2} + \frac{1-\overline{h}_1^{(t)}}{\tau_2^2}, \dots, \frac{\overline{h}_{nk}^{(t)}}{\tau_1^2} + \frac{1-\overline{h}_{nk}^{(t)}}{\tau_2^2}\Big), \quad (14)$$

$$\overline{G}^{(t)} = \mathrm{diag}\Big(\frac{\overline{g}_1^{(t)}}{\rho_1^2} + \frac{1-\overline{g}_1^{(t)}}{\rho_2^2}, \dots, \frac{\overline{g}_{km}^{(t)}}{\rho_1^2} + \frac{1-\overline{h}_{km}^{(t)}}{\rho_2^2}\Big), \quad (15)$$

and, for $p = 1, \dots, nk$, and $j = 1, \dots, km$,

$$\overline{h}_p^{(t)} = \frac{\alpha_1\mathcal{N}(v_p|0,\tau_1^2)}{\alpha_1\mathcal{N}(v_p|0,\tau_1^2) + (1-\alpha_1)\mathcal{N}(v_p|0,\tau_2^2)} \quad (16)$$

$$\overline{g}_j^{(t)} = \frac{\alpha_2\mathcal{N}(z_j|0,\rho_1^2)}{\alpha_2\mathcal{N}(z_j|0,\rho_1^2) + (1-\alpha_2)\mathcal{N}(z_j|0,\rho_2^2)}. \quad (17)$$

4.2. The M-Step

In the M-step, the parameter estimates are updated by maximizing $\mathcal{Q}(\theta, \hat{\theta}^{(t)})$ with respect to θ. Examining the several terms in (13) reveals that there are two types of problems: with respect to V and Z, we face a low-rank matrix factorization problem, in the form proposed in [42]; for the other parameters, closed-form updates can be obtained by equating the corresponding derivatives to zero.

4.2.1. Prototypes and factors

Considering only the terms in $\mathcal{Q}(\theta, \hat{\theta}^{(t)})$ that depend on V and Z, we have the following low-rank factorization problem,

$$\arg\min_{V,Z}\Big[\frac{||D-VZ||_F^2}{2\sigma^2} + \frac{1}{2}v^T\overline{H}^{(t)}v + \frac{1}{2}z^T\overline{G}^{(t)}z\Big], \quad (18)$$

which is a generalization of the recently proposed unified model proposed in [42]. Instead of the plain Frobenius norms used in [42], (18) uses weighted Frobenius norms. In fact, notice that v^TSv (where $v = \mathrm{vec}(V)$ and S is some diagonal matrix) is simply the square of a weighted version of the Frobenius norm: $v^TSv = \sum_i S_{ii}v_i^2$.

Inspired by the optimization method in [42], we tackle problem (18) via the *augmented Lagrangian method* (ALM) [43], also known as the *method of multipliers* (MM) [44,45]. The first step is to rewrite (18) as an equivalent constrained problem, via a procedure known as variable splitting (*i.e.*, introducing a new variable C to take the place of the low rank product VZ):

$$\arg\min_{V,Z,C}\Big[\frac{||D-C||_F^2}{2\sigma^2} + \frac{1}{2}v^T\overline{H}^{(t)}v + \frac{1}{2}z^T\overline{G}^{(t)}z\Big] \quad (19)$$
$$\text{s.t.}\quad C = VZ.$$

For computational purposes, it is more convenient to write a fully vectorized version of this problem; to that end (and as for $v = \mathrm{vec}(V)$ and $z = \mathrm{vec}(Z)$), we define $c = \mathrm{vec}(C)$ and $d = \mathrm{vec}(D)$, leading to

$$\arg\min_{v,z,c}\Big[\frac{||d-c||_2^2}{2\sigma^2} + \frac{1}{2}v^T\overline{H}^{(t)}v + \frac{1}{2}z^T\overline{G}^{(t)}z\Big] \quad (20)$$
$$\text{s.t.}\quad c = (I\otimes V)z,$$

where the constraint $c = (I\otimes V)z$ is equivalent to $C = VZ$ (as is clear from (1)). Notice that the constraint can also be written as $c = (Z^T\otimes I)v$ (as is also clear from (1)). For later use, we define the two following matrices:

$$A(z) = (Z^T\otimes I) \qquad \text{and} \qquad B(v) = (I\otimes V). \quad (21)$$

The augmented Lagrangian for problem (19) is obtained by adding a quadratic penalty to the Lagrange function of problem (20),

$$\begin{aligned}\mathcal{L}_\varrho(v,z,c,y) = &\frac{||d-c||^2}{2\sigma^2} + \frac{1}{2}v^T\overline{H}v + \frac{1}{2}z^T\overline{G}z\\ &+\frac{\varrho}{2}||B(v)z - c||^2 + y^T(c - B(v)z),\end{aligned} \quad (22)$$

where y is the vector of Lagrange multipliers, $\varrho\geq 0$ is a parameter, and we have written $\overline{H} = \overline{H}^{(t)}$ and $\overline{G} = \overline{G}^{(t)}$ to keep the notation lighter. Notice that the vector $B(v)z$ can also be equivalently written as $A(z)v$. The ALM proceeds by alternating between minimizing $\mathcal{L}_\varrho(v,z,c,y)$ with respect to the variables v, z, c and updating the Lagrange multipliers.

Unfortunately, $\mathcal{L}_\varrho(v,z,c,y)$ cannot be minimized in closed-form simultaneously with respect to v, z, c, thus we follow the approach in [42] and solve it by a non-linear block Gauss–Seidel (NLBGS) method, *i.e.*, we cycle through minimizations with respect to v, z, and c, until some convergence criterion is satisfied, taking advantage of the fact that each of these minimizations can be written in closed form, simply by equating the corresponding gradients to zero. Letting the iteration counter of NLBGS be s and denoting $A^{(s)} = A(z^{(s)})$ and $B^{(s)} = B(v^{(s)})$, the resulting update expressions are (for $s = 1, 2, \dots$)

$$v^{(s+1)} = \Big(\overline{H} + \varrho\big(A^{(s)}\big)^TA^{(s)}\Big)^{-1}\Big(\big(A^{(s)}\big)^Ty + \varrho\big(A^{(s)}\big)^Tc^{(s)}\Big) \quad (23)$$

$$z^{(s+1)} = \Big(\overline{G} + \varrho\big(B^{(s+1)}\big)^TB^{(s+1)}\Big)^{-1}\Big(\big(B^{(s+1)}\big)^Ty + \varrho\big(B^{(s+1)}\big)^Tc^{(s)}\Big) \quad (24)$$

$$c^{(s+1)} = \frac{d - \sigma^2y + \varrho B^{(s+1)}z^{(s+1)}}{1 + \sigma^2\varrho}. \quad (25)$$

In summary, the updated $V^{(t+1)}$ and $Z^{(t+1)}$, which are the solutions of problem (18), are obtained by cycling through (23), (24), and (25), until some convergence criterion is satisfied.

4.2.2. Other parameters

The update of other parameters $(\tau_1^2, \tau_2^2, \rho_1^2, \rho_2^2, \sigma, \alpha_1, \alpha_2)$ are obtained by setting the corresponding partial derivatives of $\mathcal{Q}(\theta, \hat{\theta}^{(t)})$ to zero, yielding the following expressions:

$$\tau_1^2 = \big(v^T\overline{H}v\big)/\|\overline{H}\|_F \quad (26)$$

$$\tau_2^2 = v^T(1-\overline{H})v/\|1-\overline{H}\|_F \quad (27)$$

$$\rho_1^2 = z^T\overline{G}z/\|\overline{G}\|_F \quad (28)$$

$$\rho_2^2 = z^T(1-\overline{G})z/\|1-\overline{G}\|_F \quad (29)$$

$$\alpha_1 = \left(\sum_{p=1}^{nk}\overline{h}_p\right)/(nk) \quad (30)$$

$$\alpha_2 = \left(\sum_{p=1}^{nk}\overline{g}_p\right)/(mk) \quad (31)$$

$$\sigma^2 = \|D-VZ\|_F^2/(nm), \quad (32)$$

where we have omitted the iteration counter superscript $(\cdot)^{(t)}$, to keep the notation lighter.

4.3. The complete algorithm

The final complete algorithm obtained by putting together the E-step and M-step derived in the previous subsections is presented in Algorithm 1. Some comments and explanations about the algorithm are in order, and are presented in the next few paragraphs.

Algorithm 1 SSBiEM.

```
Require: Data matrix $D$, number of biclusters $k$.
 1: Initialize $V, Z$ using TSVD($k$)
 2: Initialize $\tau_1^2, \tau_2^2, \rho_1^2, \rho_2^2, \alpha_1, \alpha_2, \sigma^2$ (see text)
 3: Initialize $v \leftarrow$ vec($V$) and $z \leftarrow$ vec($Z$)
 4: while EM not converged do
      E-Step:
 5:     compute $\overline{H}$ and $\overline{G}$ using (14), (15), (16), (17)
 6:     $B \leftarrow I \otimes V$, where $V \leftarrow \text{vec}^{-1}(v)$
 7:     $A \leftarrow Z^T \otimes I$, where $Z \leftarrow \text{vec}^{-1}(z)$
 8:     $c \leftarrow Bz$
      M-Step:
 9:     while ALM not converged do
10:       while NLBGS not converged do
11:         Update $v$ according to (23)
12:         $B \leftarrow I \otimes V$, where $V \leftarrow \text{vec}^{-1}(v)$
13:         Update $z$ according to (24)
14:         $A \leftarrow Z^T \otimes I$, where $Z \leftarrow \text{vec}^{-1}(z)$
15:         Update $c$ according to (25)
16:       end while
17:       $y \leftarrow y + \varrho(c - Bz)$
18:       $\varrho \leftarrow \min(\varrho\mu, 10^{20})$
19:     end while
20:     update parameters according to (26)–(32)
21: end while
22: return $V, Z, \overline{H}, \overline{G}$
```

Initialization is carried out in lines 1 and 2, where TSVD(k) stands for the *k-truncated singular value decomposition*, which corresponds to computing the SVD of D and keeping only the left and right singular vector corresponding to the k largest singular values (this is known to correspond to the best rank k approximation of D in the Frobenious norm sense). The other parameters are initialized as follows: σ^2 is initialized according to (32), using the initial V and Z; α_1 and α_2 are initialized to 1/2; finally, the spike and slab variances τ_1^2 and ρ_1^2 are initialized as the standard deviation of V and Z respectively, and τ_2^2 and ρ_2^2 are set to one tenth of τ_1^2 and ρ_1^2. Line 5 corresponds to the E-step of the EM algorithm, as explained in Section 4.1. Lines 6, 7, and 8 are the initialization of the ALM method described in Section 4.2.1. The inner loop of the NLBGS algorithm that implements the update step (with respect to v, z, and c) of ALM is implemented in lines 10–16; the update of the Lagrange multipliers y is implemented in line 17. As in [42], the ALM parameter ϱ is increased at each interation, by multiplying it by $\mu = 1.05$ in line 18. Finally the remaining model parameter estimates are updated according to (26)–(32), in line 20, completing the M-step.

It is important to stress that the SSBi model and the SSBiEM algorithm herein presented can be trivially generalized to the case where each bicluster has its own spike and slab parameters; instead of a common set $\tau_1^2, \tau_2^2, \rho_1^2, \rho_2^2, \alpha_1, \alpha_2$, each bicluster (*i.e.*, each of the k rows of V and columns of Z) will have its own set of parameters, resulting in a more complicated (but essentially equivalent) set of update equations. To keep the notation simpler, we abstained from presenting that more general version of the model and algorithm, but it was used in the experiments reported below.

Since the complete algorithm includes 3 nested loops (EM, ALM, NLBGS), it involves three stopping criteria (lines 4, 9, and 10). The EM stopping criterion is based on the relative change of the log-likelihood function falling below some threshold. The ALM iterations stop when the relative change in the Lagrange multiplier vector y is less than a threshold. Finally, the inner NLBGS loop is stopped when the maximum of relative changes in the involved variables is below a threshold.

Of course, the EM algorithm proposed in this paper involves an approximate M-Step (hence being a *generalized EM* algorithm [40,46]), where an iterative procedure minimizes a non-convex function, thus there are no formal guarantees of convergence and the results may depend on the initialization.[2] However, in all the experiments discussed in the following section, SSBiEM always converged to an effective solution.

Concerning space complexity, the leading term is $\mathcal{O}(nk)$ (or $\mathcal{O}(km)$, depending on the maximum between the number of rows/columns in the original data matrix) which is the space needed to store the $A(z)$ (or $B(v)$) matrix. Notice that, even if the size of those matrices grows with the number of rows/columns of the original data matrix, these matrices are mostly zeros (since they involve a Kronecker product with an identity matrix). Thus, an adequate sparse representation can overcome this possible drawback. Regarding time complexity (for each iteration), the leading term is $\mathcal{O}(n^3k^3)$ (or $\mathcal{O}(m^3k^3)$) which is the worst case scenario for matrix multiplication/inversion of a $\mathcal{O}(nk)$ (or $\mathcal{O}(mk)$) matrix. In all the experiments presented in the next section, the proposed EM algorithm, on average, converged in $\sim$14.6 iterations, whereas NLGBS and ALM required, respectively, $\sim$3.6 and $\sim$61 iterations.

5. Experimental evaluation

In this section, the SSBiEM algorithm[3] is experimentally evaluated on both synthetic and real datasets.

[2] Notice that, although the convergence stndard EM is guaranteed with a regular exponential family, in this case we would need to have the guarantee that the ALM/NLBGS algorithm increases the Q-function. Unfortunately, we have no formal guarantee that the ALM/NLBGS satisfies that condition, although in the experiments this was also te case.

[3] SSBiEM is available as Matlab function (.m) at https://github.com/emme-di/SSBiEM/.

Table 1
Synthetic Dataset Results. The table compares state-of-the-art approaches on the synthetic benchmark dataset proposed in [20]. Other approaches results have been taken from Hochreiter et al. [20].

Method	Score	References
SSBi	**0.606**	
FABIAS	0.564	[20]
FABIA	0.478	[20]
MFSC	0.057	[48]
plaid_ss	0.045	[49]
plaid_ms	0.072	[49]
plaid_ms_5	0.083	[49]
ISA_1	0.333	[50]
ISA_2	0.299	[50]
ISA_3	0.188	[50]

Table 2
Breast Tumor Dataset Results. The table shows the results on the real Breast Tumor gene expression dataset taken from [7]. The GO results for the other approaches have been taken from Mukhopadhyay et al. [7].

Method	Score (%)	References
FABIA	55	[20]
ISA	63	[50]
Hierarc.	70	[56]
SAMBA	73	[23]
FLOC	85	[12]
OOB	87.5	[55]
SSBi	**87.5**	

5.1. *Synthetic benchmark*

To obtain a fair comparison and a clear perspective on the performance of SSBi with respect to the FABIA approach, we carry out experiments on the synthetic benchmark datasets proposed by Hochreiter *et al* [20], and adopt their evaluation criteria. The dataset is composed by 100 matrices of dimension 1000×100, simulating real world gene expression datasets. Each matrix contains 10 implanted biclusters, generated with a multiplicative structure, where the positions and dimensions were randomly chosen, thus we run SSBiEM with $k = 10$. For a given set of true biclusters T and a set of retrieved biclusters B, the accuracy of B with the following three steps [20]:

1. compute the *Jaccard similarity coefficient* $J(t, b)$ of all the pairs $(t, b) \in T \times B$; notice that $J(t, b) \in [0, 1]$, with $J(t, b) = 0$, if $t \cap b = \emptyset$, and $J(t, b) = 1$, if $t = b$;
2. via the *Kuhn–Munkres algorithm* (a.k.a. the*Hungarian algorithm* [47]), assign each bicluster in T to one in B, by maximizing the sum of the Jaccard similarities of the assigned pairs;
3. divide the resulting assignment value by $\max\{|T|, |B|\}$; the final result is a quantity in $[0, 1]$, which is equal to 1 if and only if $B = T$.

The results are shown in Table 1, where we compared SSBiEM with two versions of FABIA and with other state-of-the-art methods (the results of FABIA and of the other methods are those reported in [20]). SSBiEM outperforms all the other methods on this dataset, proving its effectiveness.

5.2. *Real datasets*

The SSBiEM algorithm was also compared with other state-of-the-art methods on two real dataset. The first one is a classical dataset used in biclustering experiments: the Breast Tumor microarray gene expression dataset [7]. Since FABIA is the closest formulation to SSBi, we decided to test SSBi on a dataset where FABIA does not perform well, to assess if the novelties introduced in SSBi with respect to FABIA provide a significant improvement. On the other hand, FABIA has been recently applied with success in the *multiple structure recovery* (MSR) task [51]; thus, we also decided to assess how SSBiEM perfors on this task, and if it competitive with FABIA.

5.2.1. *Gene expression dataset*

Following a standard approach in the biclustering literature, the performance on gene expression matrices is assessed through a biological validation, by analyzing *Gene Onthology* (GO) terms [2,7]. Such validation indicates how significantly the set of genes belonging to a retrieved bicluster is enriched by a GO category, as provided by GO Consortium [7,52]. The analysis of these terms is performed automatically using the FuncAssociate web-server (http://llama.mshri.on.ca/funcassociate/), which provides a score, at a given significance level, that corresponds to the percentage of gene sets that are enriched with respect to at least one GO annotation. In our case we use this procedure setting the significance level to 5%.

As also commonly proposed in the literature [53–55], we applied a variance-based gene selection procedure to reduce the dataset dimensionality, keeping 2500 genes. For a fair comparison, in [7] the authors selected the same number of biclusters for each method (40). For our algorithm, since the background of the data matrix is not zero, we run the SSBiEM algorithm with the number of biclusters set to 41. This provides a bicluster that accounts for the background noise of the data matrix, containing all the rows and columns thereof. In the end, that background bicluster is discarded to obtain the pool of 40 biclusters needed to assess the method.

The results are shown in Table 2 (the scores of the other methods are those reported in [7]); it is clear that the proposed SSBi approach compares favorably with the other state-of-the-art methods.

5.2.2. *Multiple structure recovery dataset*

Multiple structure recovery (MSR) concerns the extraction of multiple models from noisy or outlier-contaminated data. MSR is an important and challenging problem, which emerges in many computer vision applications [57–59]. In general, an instance of an MSR problem is represented by a *preference matrix* containing, in one dimension, the points under analysis, and in the other, the hypotheses/structures to which points should belong. The entry (i, j) in this matrix indicates how well a certain point i is represented by the given hypothesis/structure j.

The Adelaide dataset, where FABIA has been recently applied, involves two type of MSR problems: motion and plane estimation. Given two different images of the same scene, where several objects move independently, motion segmentation aims at recovering subsets of point matches that undergo the same motion. Given two uncalibrated views of a scene, plane segmentation consists in retrieving the multi-planar structures by fitting homographies to point correspondences. The AdelaideRMF dataset[4] is composed of 38 image pairs (19 for motion segmentation and 19 for plane segmentation), with matching points contaminated by strong outliers. The ground-truth segmentations are also available. As in [51,60], we adopt the misclassification errors to assess the results. For fair comparison, we adopt the same preference matrices fed to FABIA, which were generated as presented in [61].

Table 3 presents the results. We report three different results for FABIA; we run the algorithm varying the parameters in the sug-

[4] https://cs.adelaide.edu.au/~hwong/doku.php?id=data.

Table 3
Misclassification error (ME %) for motion segmentation (above) and planar segmentation (below). *k* is the number of models and % out is the percentage of outliers.

	k	%out	FABIA best	FABIA best set	FABIA automatic	SSBiEM
biscuitbookbox	3	37.21	3.86	4.17	62.55	8.65
breadcartoychips	4	35.20	4.2	7.76	65.40	17.89
breadcubechips	3	35.22	0.87	0.87	64.78	7.74
breadtoycar	3	34.15	0.60	0.72	66.27	5.90
carchipscube	3	36.59	1.52	1.70	63.64	8.36
cubebreadtoychips	4	28.03	1.07	9.79	73.09	11.07
dinobooks	3	44.54	9.72	10.44	56.94	20.83
toycubecar	3	36.36	9.50	25.70	64.00	10.80
biscuit	1	57.68	0	19.27	44.24	2.00
biscuitbook	2	47.51	1.32	1.58	52.49	3.93
boardgame	1	42.48	8.96	9.10	59.50	19.64
book	1	44.32	0	29.20	56.15	1.50
breadcube	2	32.19	19.42	20.66	68.18	5.12
breadtoy	2	37.41	19.62	19.65	63.19	0.97
cube	1	69.49	1.66	7.22	32.12	5.30
cubetoy	2	41.42	2.21	7.87	60.24	3.13
game	1	73.48	0	0.34	27.04	5.75
gamebiscuit	2	51.54	2.44	2.56	49.09	5.98
cubechips	2	51.62	0.53	0.85	49.65	4.30
mean			4.61	9.45	49.23	7.84
median			1.66	7.76	60.24	5.90
	k	%out	FABIA best	FABIA best set	FABIA automatic	SSBiEM
unionhouse	5	18.78	21.54	38.01	23.49	23.49
bonython	1	75.13	6.82	8.69	26.26	17.47
physics	1	46.60	0.00	32.26	54.72	10.38
elderhalla	2	60.75	3.04	4.77	39.25	30.37
ladysymon	2	33.48	11.81	41.43	67.51	22.36
library	2	56.13	20.47	27.81	44.65	44.65
nese	2	30.29	4.92	14.80	66.54	2.91
sene	2	44.49	2.20	4.96	52.80	5.20
napiera	2	64.73	21.85	35.36	37.09	44.04
hartley	2	62.22	23.59	40.81	38.44	42.06
oldclassicswing	2	32.23	7.92	24.22	67.55	13.25
barrsmith	2	69.79	29.88	54.69	31.12	64.81
neem	3	37.83	11.20	23.49	63.49	38.42
elderhallb	3	49.80	18.63	34.27	52.16	38.67
napierb	3	37.13	36.68	39.54	60.62	40.62
johnsona	4	21.25	17.96	19.89	79.09	25.42
johnsonb	7	12.02	24.50	43.57	87.98	48.94
unihouse	5	18.78	15.76	26.07	83.45	29.02
bonhall	6	6.43	24.02	53.03	93.82	53.09
mean			15.94	29.88	50.93	31.33
median			17.96	32.26	54.72	30.37

gested range as in [20,51], and on the basis of the considered results the performances can vary significantly. The fourth columns of Table 3 (*FABIA best*) shows the results provided in [51] with FABIA: in this case we consider for each different matrix the best performance with respect to the misclassification error. The results in the fifth column (*FABIA best set*), which are slightly worse than the previous, are obtained by selecting the best set of parameters values minimizing the misclassification error (one for the motion segmentation and one for the plane estimation). The second-to-last column (*FABIA automatic*) shows the misclassification performances where biclusters have been selected by adopting the quality measures provided by the FABIA algorithm. In fact, FABIA provides a score for each bicluster retrieved indicating its amount of information. To obtain the *FABIA automatic* column we kept, for each matrix, the set of biclusters with the highest average of such scores. Finally the last columns of the table represent the results obtained with the SSBiEM approach. The table show that FABIA performances significantly vary with respect to the chosen set of parameters.

In contrast, SSBiEM automatically learns all these parameters directly from the data matrix. Table 3 shows that our approach outperforms *FABIA automatic* and favourably compares with *FABIA best set* columns, which we think represent a fair comparison. Although *FABIA best* retrieves better results, in this case we would set FABIA parameters manually for every different matrix, whereas SSBiEM computes them automatically.

6. Conclusion and discussion

In this paper, we proposed *spike and slab biclustering* (SSBi), a novel probabilistic generative model for biclustering based on a sparse low-rank matrix factorization perspective. In more detail, the method involves the factorization of the data matrix into two matrices containing information about the underlying biclusters; these matrices are encouraged to be sparse, modeling the fact that, in most applications, the data matrix is much bigger than the biclusters. The sparsity-inducing nature of the model is embodied in a spike and slab formulation. To estimate the parameters of proposed model, we proposed an instance of the EM algorithm, termed SSBiEM. At each iteration of SSBiEM, a low-rank matrix factorization problem is solved, exploiting a recently proposed formulation and augmented Lagrangian algorithm. An important feature of the SSBi approach is that it provides information about the bicluster membership, without the need for any post-processing. Finally, the proposed method was compared with state-of-the-art approaches, on both synthetic and real datasets, and the results obtained by SSBi compare favorably with those of the other methods.

References

[1] Y. Cheng, G. Church, Biclustering of expression data, in: Proc. Eighth Int. Conf. on Intelligent Systems for Molecular Biology (ISMB00), 2000, pp. 93–103.
[2] S. Madeira, A. Oliveira, Biclustering algorithms for biological data analysis: a survey., IEEE Trans. Comput. Biol. Bioinf. 1 (2004) 24–44.
[3] A. Oghabian, S. Kilpinen, S. Hautaniemi, E. Czeizler, Biclustering methods: biological relevance and application in gene expression analysis, PloS one 9 (3) (2014) e90801.
[4] D. Yan, J. Wang, Biclustering of gene expression data based on related genes and conditions extraction, Pattern Recognit. 46 (4) (2013) 1170–1182.
[5] N. Gupta, S. Aggarwal, Mib: using mutual information for biclustering gene expression data, Pattern Recognit. 43 (8) (2010) 2692–2697.
[6] S. Dolnicar, S. Kaiser, K. Lazarevski, F. Leisch, Biclustering overcoming data dimensionality problems in market segmentation, J. Travel Res. 51 (1) (2012) 41–49.
[7] A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay, C.A.C. Coello, Survey of multi-objective evolutionary algorithms for data mining: part ii, Evol. Comput., IEEE Trans. 18 (1) (2014) 20–35.
[8] P. de Castro, F. de França, H. Ferreira, F. Von Zuben, Applying biclustering to text mining: an immune-inspired approach, Artif. Immune Syst. (2007) 83–94.
[9] A.A. Irissappane, S. Jiang, J. Zhang, A biclustering-based approach to filter dishonest advisors in multi-criteria e-marketplaces, in: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 1385–1386.
[10] M. Kaytoue, V. Codocedo, A. Buzmakov, J. Baixeries, S.O. Kuznetsov, A. Napoli, Pattern structures and concept lattices for data mining and knowledge processing, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2015, pp. 227–231.
[11] E. Savia, K. Puolamäki, S. Kaski, Latent grouping models for user preference prediction, Mach. Learn. 74 (1) (2009) 75–109.
[12] J. Yang, H. Wang, W. Wang, P.S. Yu, An improved biclustering method for analyzing gene expression profiles, Int. J. Artif. Intell. Tools 14 (05) (2005) 771–789.
[13] T. Murali, S. Kasif, Extracting conserved gene expression motifs from gene expression data., in: Pacific Symposium on Biocomputing, 8, World Scientific, 2003, pp. 77–88.
[14] H. Wang, W. Wang, J. Yang, P.S. Yu, Clustering by pattern similarity in large data sets, in: Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, ACM, 2002, pp. 394–405.
[15] S. Yoon, C. Nardini, L. Benini, G. De Micheli, Discovering coherent biclusters from gene expression data using zero-suppressed binary decision diagrams, IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB) 2 (4) (2005) 339–354.
[16] G. Getz, E. Levine, E. Domany, Coupled two-way clustering analysis of gene microarray data, Proc. Natl. Acad. Sci. U S A 97 (22) (2000) 12079–12084.
[17] A. Farinelli, M. Denitto, M. Bicego, Biclustering of expression microarray data using affinity propagation, in: M. Loog, L. Wessels, M. Reinders, D. Ridder (Eds.), Pattern Recognition in Bioinformatics, Lecture Notes in Computer Science, 7036, Springer Berlin Heidelberg, 2011, pp. 13–24, doi:10.1007/978-3-642-24855-9_2.
[18] J.A. Hartigan, Direct clustering of a data matrix, J. Am. Stat. Assoc. 67 (337) (1972) 123–129.
[19] C. Tang, L. Zhang, A. Zhang, M. Ramanathan, Interrelated two-way clustering: an unsupervised approach for gene expression data analysis, in: Bioinformatics and Bioengineering Conference, 2001. Proceedings of the IEEE 2nd International Symposium on, IEEE, 2001, pp. 41–48.
[20] S. Hochreiter, U. Bodenhofer, M. Heusel, A. Mayr, A. Mitterecker, A. Kasim, T. Khamiakova, S. Van Sanden, D. Lin, W. Talloen, et al., Fabia: factor analysis for bicluster acquisition, Bioinformatics 26 (12) (2010) 1520–1527.
[21] D.J. Reiss, N.S. Baliga, R. Bonneau, Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks, BMC Bioinf. 7 (1) (2006) 280.
[22] Q. Sheng, Y. Moreau, B. De Moor, Biclustering microarray data by gibbs sampling, Bioinformatics 19 (suppl 2) (2003) ii196–ii205.
[23] A. Tanay, R. Sharan, R. Shamir, Discovering statistically significant biclusters in gene expression data, Bioinformatics 18 (suppl 1) (2002) S136–S144.
[24] H. Zhao, K.L. Chan, L.-M. Cheng, H. Yan, A probabilistic relaxation labeling framework for reducing the noise effect in geometric biclustering of gene expression data, Pattern Recognit. 42 (11) (2009) 2578–2588.
[25] A. Tanay, R. Sharan, R. Shamir, Biclustering algorithms: a survey, 2004,.
[26] J. Hardin, J. Wilson, A note on oligonucleotide expression values not being normally distributed, Biostatistics (2009) kxp003.
[27] K. Bunte, E. Leppaho, I. Saarinen, S. Kaski, Sparse group factor analysis for biclustering of multiple data sources, Bioinformatics 32 (16) (2016) 2457–2463.
[28] T.J. Mitchell, J.J. Beauchamp, Bayesian variable selection in linear regression, J. Am. Stat. Assoc. 83 (404) (1988) 1023–1032.
[29] H. Ishwaran, J.S. Rao, Spike and slab variable selection: frequentist and bayesian strategies, Ann. Stat. (2005) 730–773.
[30] J. Magnus, H. Neudecker, Matrix Differential Calculus with Applications in Statistics and Econometrics, Wiley, 1999.
[31] M. Lee, H. Shen, J.Z. Huang, J. Marron, Biclustering via sparse singular value decomposition, Biometrics 66 (4) (2010) 1087–1095.
[32] J. Yoo, S. Choi, Orthogonal nonnegative matrix tri-factorization for co-clustering: multiplicative updates on stiefel manifolds, Inf. Process. Manag. 46 (5) (2010) 559–570.
[33] H. Wang, F. Nie, H. Huang, F. Makedon, Fast nonnegative matrix tri-factorization for large-scale data co-clustering, in: IJCAI Proceedings-International Joint Conference on Artificial Intelligence, 22, 2011, p. 1553.
[34] T. Li, C. Ding, The relationships among various nonnegative matrix factorization methods for clustering, in: Data Mining, 2006. ICDM'06. Sixth International Conference on, IEEE, 2006, pp. 362–371.
[35] G. Govaert, M. Nadif, An em algorithm for the block mixture model, IEEE Trans. Pattern Anal. Mach. Intell. 27 (4) (2005) 643–647.
[36] D. Vu, M. Aitkin, Variational algorithms for biclustering models, Comput. Stat. Data Anal. 89 (2015) 12–24.
[37] S. Pledger, R. Arnold, Multivariate methods using mixtures: correspondence analysis, scaling and pattern-detection, Comput. Stat. Data Anal. 71 (2014) 241–261.
[38] M. Ailem, F. Role, M. Nadif, Sparse poisson latent block model for document clustering, IEEE Trans. Knowl. Data Eng. (2017).
[39] M. Girolami, A variational method for learning sparse and overcomplete representations, Neural Comput. 13 (11) (2001) 2517–2532.
[40] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. Ser. B 39 (1977) 1–38.
[41] D.S. Cheng, V. Murino, M. Figueiredo, Clustering under prior knowledge with application to image segmentation, in: P.B. Schölkopf, J.C. Platt, T. Hoffman (Eds.), Advances in Neural Information Processing Systems 19, MIT Press, 2007, pp. 401–408.
[42] R. Cabral, F. De la Torre, J.P. Costeira, A. Bernardino, Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition, in: Computer Vision (ICCV), 2013 IEEE International Conference on, IEEE, 2013, pp. 2488–2495.
[43] J. Nocedal, S. Wright, Numerical Optimization, Springer, 2006.
[44] M. Hestenes, Multiplier and gradient methods, J. Optim. Theory Appl. 4 (1969) 303–320.
[45] M. Powell, A method for nonlinear constraints in minimization problems, in: R. Fletcher (Ed.), Optimization, Academic Press, 1969, pp. 283–298.
[46] C.J. Wu, On the convergence properties of the em algorithm, Ann. Stat. (1983) 95–103.
[47] J. Munkres, Algorithms for the assignment and transportation problems, J. Soc. Indus. Appl. Math. 5 (1) (1957) 32–38.
[48] P.O. Hoyer, Non-negative matrix factorization with sparseness constraints, J. Mach. Learn. Res. 5 (2004) 1457–1469.
[49] L. Lazzeroni, A. Owen, et al., Plaid models for gene expression data, Stat. Sin. 12 (1) (2002) 61–86.
[50] J. Ihmels, S. Bergmann, N. Barkai, Defining transcription modules using large-scale gene expression data, Bioinformatics 20 (13) (2004) 1993–2003.
[51] M. Denitto, L. Magri, A. Farinelli, A. Fusiello, M. Bicego, Multiple structure recovery via probabilistic biclustering, in: Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Springer, 2016, pp. 274–284.
[52] M. Ashburner, C.A. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, et al., Gene ontology: tool for the unification of biology, Nat. Genet. 25 (1) (2000) 25–29.
[53] M. Bicego, P. Lovato, A. Perina, M. Fasoli, M. Delledonne, M. Pezzotti, A. Polverari, V. Murino, Investigating topic models' capabilities in expression microarray data classification, IEEE/ACM Trans. Comput. Biolo. Bioinf. (TCBB) 9 (6) (2012) 1831–1836.
[54] S. Rogers, M. Girolami, C. Campbell, R. Breitling, The latent process decomposition of cdna microarray data sets, IEEE/ACM Trans. Comput. Biol. Bioinf. 2 (2) (2005) 143–156.
[55] M. Denitto, A. Farinelli, M. Bicego, Biclustering gene expressions using factor graphs and the max-sum algorithm, in: Proceedings of the 24th International Conference on Artificial Intelligence, AAAI Press, 2015, pp. 925–931.
[56] R.R. Sokal, A statistical method for evaluating systematic relationships, Univ. Kans. Sci. Bull. 38 (1958) 1409–1438.
[57] A.W. Fitzgibbon, A. Zisserman, Multibody structure and motion: 3-d reconstruction of independently moving objects, in: Computer Vision-ECCV 2000, Springer, 2000, pp. 891–906.
[58] C. Häne, C. Zach, B. Zeisl, M. Pollefeys, A patch prior for dense 3d reconstruction in man-made environments, in: 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on, IEEE, 2012, pp. 563–570.
[59] R. Toldo, A. Fusiello, Image-consistent patches from unstructured points with j-linkage, Image Vis. Comput. 31 (10) (2013) 756–770.
[60] M. Soltanolkotabi, E. Elhamifar, E.J. Candès, Robust subspace clustering, Ann. Statist. 42 (2) (2014) 669–699.
[61] L. Magri, A. Fusiello, Robust multiple model fitting with preference analysis and low-rank approximation, in: X. Xie, M.W. Jones, G.K.L. Tam (Eds.), Proceedings of the British Machine Vision Conference (BMVC), BMVA Press, 2015, pp. 20.1–20.12, doi:10.5244/C.29.20.

Matteo Denitto is a Ph.D. Student at the University of Verona. His research interests involve biclustering, graphical model and factor graphs. Matteo Denitto received his bachelor and master degree in Bioinformatics from the University of Verona in 2011 and 2013 respectively.

Alessandro Farinelli is Associate Professor at University of Verona, Computer Science Department. His research interests comprise theoretical and practical issues related to the development of Artificial Intelligent Systems applied to robotics. He participated in several national and international research projects in the broad area of Artificial Intelligence for robotic systems.

Mario A.T. Figueiredo received E.E., M.Sc., and Ph.D. degrees in electrical and computer engineering from Instituto Superior Tecnico, University of Lisbon, where he is now a full Professor. His interests include statistical pattern recognition, machine learning, and computer vision. He is a Fellow of the IEEE and of the IAPR.

Manuele Bicego is an Assistant Professor at the University of Verona. His research interests include pattern recognition, bioinformatics and computer vision. He actively participated to different private, national and EU-funded research projects. He served as Guest Editor of the special issue of the Pattern Recognition journal on “Similarity-based Pattern Recognition”.